```
+-----------------------------------+
|          Quiz Application          |
+-----------------------------------+
|                                 |
|       +-----------------+        |
|       |  User Interface|        |
|       +-----------------+        |
|               |               |
|               |               |
|       +-----------------+        |
|       |   JavaScript   |        |
|       |   Application  |        |
|       +-----------------+        |
|               |               |
|               |               |
|       +-----------------+        |
|       |  Quiz Logic    |        |
|       +-----------------+        |
|               |               |
|               |               |
|       +-----------------+        |
|       |   Data Store   |        |
|       +-----------------+        |
|                                 |
+-----------------------------------+
```

Explanation of the Architecture:

User Interface:

Responsible for presenting the quiz questions and answer options to the user.

Captures user input (selected answers) and triggers events based on user actions.

Renders the user's score and other relevant information.

JavaScript Application:

Handles the dynamic loading of quiz questions and answer options from the data store.

Manages the user's interaction and handles the submission of answers.

Calculates and updates the user's score based on the selected answers.

Controls the flow of the quiz by loading the next question and managing the quiz state.

Quiz Logic:

Contains the core logic and rules of the quiz application.

Implements functions for loading questions, calculating scores, and managing the quiz state.

Validates user inputs and ensures the correct functioning of the application.

Communicates with the data store to retrieve quiz questions and answers.

Data Store:

Stores the quiz questions and answer options.

Provides methods to retrieve the quiz data for the JavaScript application.

Can be implemented as an array of objects, a database, or any other suitable storage solution.

The architecture follows a modular approach, with each component having specific responsibilities. The User Interface interacts with the user, the JavaScript Application manages user inputs and quiz flow, the Quiz Logic implements the quiz rules, and the Data Store stores and provides the quiz data.

This architecture promotes separation of concerns, making the application easier to maintain, test, and enhance. It allows for scalability and flexibility in terms of adding more features, expanding the data store, or modifying the user interface while keeping the core functionality intact.